Economia com OpenBSD + PF + CARP

Anderson Nadal
http://web.onda.com.br/nadal

Humberto Sartini
http://web.onda.com.br/humberto

Palestrante

Anderson Nadal

- Coordenador Técnico do Provedor OndaRPC
- Mantenedor do VDA-Postfix
  http://web.onda.com.br/nadal
- Participante dos projetos:
  - GnuRadius
    http://www.gnu.org/software/radius/
  - Rau-Tu Redes Wan
    http://www.rau-tu.unicamp.br/redeswan/

Palestrante

Humberto Sartini

- Analista de Segurança do Provedor OndaRPC
- Participante dos projetos:
  - Rau-Tu Linux (http://www.rau-tu.unicamp.br/linux)
  - HoneypotBR (http://www.honeypot.com.br/)
  - RootCheck (http://www.ossec.net/rootcheck/)
- Participante do:
  - IV e V Fórum Internacional de SL
  - Conferência Internacional de SL (Curitiba - 2003)
  - Conisli (São Paulo/SP - 2004)

Tópicos

- História do OpenBSD
- Evolução do OpenBSD
- Características do OpenBSD
- Filtro de Pacotes (PF)
- Common Address Redundancy Protocol (CARP)
- Quality Of Services (QoS)
- Virtual Private Network (VPN)
- Estudo de Caso
- Demonstração

História do OpenBSD

- Por volta de 1970 várias licenças do Unix (AT&T) foram doadas para algumas universidades norte americanas

- Em Berkeley passaram a utilizar o Unix da AT&T, que provia o código fonte do sistema

- O único problema era a restrição que a licença impunha

História do OpenBSD

- Criadores do Unix, programadores e a universidade formaram o grupo CSRG (Computer Systems Research Group) para adicionar extensões e novidades ao Unix

- Houve a adição do TCP/IP diretamente no kernel do sistema operacional, controle de processos, memória virtual, sistemas de arquivos novos e outros recursos

História do OpenBSD

- Tecnicamente BSD é o nome dado aos avanços e melhorias do sistema operacional UNIX realizado pelo grupo CSRG

- Na versão BSD Network Release 2 muitas partes do kernel eram próprias e faltavam somente 6 arquivos para ser um sistemas operacional completo

História do OpenBSD

- Lançamento do 386/BSD, um sistema totalmente compilado e inicializado na arquitetura i386

- Surgiu o grupo NetBSD para guardar, organizar, ajudar a manter o sistema, além de liberar novas releases

- Atualmente é um grupo formado por pessoas altamente técnicas e onde o refinamento do código e a portabilidade (mais de 50 plataformas) são os aspectos principais.

História do OpenBSD

- O grupo FreeBSD formou-se poucos meses depois do NetBSD e tem como objetivo o suporte a arquitetura i386

- Elaborou processos de instalações facilitadas, distribuindo o sistema em CD-ROM e disponibilizando livremente na Internet. O foco principal é na usuabilidade do sistema e performance

História do OpenBSD

- Em meados dos anos 90, o OpenBSD se formou a partir de um “fork” do NetBSD por divergências nas políticas de segurança e modelo de desenvolvimento

- Theo de Raadt, atual líder, dividiu o NetBSD em OpenBSD no dia 18 de outubro de 1995, as 08:37, quando fez o primeira ramo de desenvolvimento no CVS da árvore do NetBSD

História do OpenBSD

- A primeira release foi a 2.0, em 1996

- No caso do OpenBSD releases são as liberações feitas a cada período de desenvolvimento, ou seja, de seis em seis meses

- A versão atual 3.7 foi liberada em 19/05/05 e a próxima está prevista para 01/11/2005.

Evolução do OpenBSD

- OpenBSD 3.0 - E-Reailed (OpenBSD Mix)

Evolução do OpenBSD

- OpenBSD 3.0- E-Reailed (OpenBSD Mix)
  - OpenSSH Protocolo 1 e 2
  - Mudanças na documentação
  - Ports mais completo
  - Novo Filtro de Pacotes PF
  - Mais de 1000 pacotes pré compilados

Evolução do OpenBSD

- OpenBSD 3.1 - Systemagic

Evolução do OpenBSD

- OpenBSD 3.1 - Systemagic
  - Melhoras do PF
  - AuthPF
  - Suporte a RAID
  - Bridge Wavelan
  - Mais de 1000 pacotes pré compilados

Evolução do OpenBSD

- OpenBSD 3.2 - GoldFlipper

Evolução do OpenBSD

- OpenBSD 3.2 - GoldFlipper
  - Apache Chroot por default
  - Encriptação de Hardware Simétrico e Assimétrico
  - Systrace
  - Ferramentas contra possíveis Buffer Overflow
  - Mais de 1800 pacotes pré compilados

Evolução do OpenBSD

- OpenBSD 3.3 – Puff the Barbarian

Evolução do OpenBSD

- OpenBSD 3.3 – Puff the Barbarian
  - ProPolice
  - W^X (W xor X)
  - Servidor X e Console X com mais segurança e privilégio separado
  - PF com novas melhorias
  - Mais de 2000 pacotes pré-compilados

Evolução do OpenBSD

- OpenBSD 3.4 – The Legend of Puffy Hood

Evolução do OpenBSD

- OpenBSD 3.4 – The Legend of Puffy Hood
  - Syslog com privilégios separados
  - strcpy, strcat, sprintf, vsprintf
  - Melhorias no ProPolices
  - Suporte a novos hardwares
  - Suporte Ready Only NTFS
  - Mais de 2400 pacote pré compilados

Evolução do OpenBSD

- OpenBSD 3.5 – Carp License and Redundancy must be free

Evolução do OpenBSD

- OpenBSD 3.5 – Carp License and Redundancy must be free
  - Novas Plataformas Amd64, Cats e mvme88k
  - Otimização no PF
  - Carp e PFSync
  - BGPD
  - Mais de 2500 pacotes pré compilados

Evolução do OpenBSD

- OpenBSD 3.6 – Pond-erosa Puff (live)

Evolução do OpenBSD

- OpenBSD 3.6 – Pond-erosa Puff (live)
  - Suporte SMP
  - OpenNTPD
  - Melhorias NFS
  - OpenSSH 3.9
  - Suporte a novos hardware
  - Mais de 2005 pacotes pré compilados

Evolução do OpenBSD

- OpenBSD 3.7 – Wizard of OS

Evolução do OpenBSD

- OpenBSD 3.7 – Wizard of OS
  - Novas plataformas Zaurus e SGI
  - Suporte a diversos USB e Wireless
  - OSPFD
  - OpenSSH 4.1
  - Mais de 3000 pacotes pré compilados

Evolução do OpenBSD

- OpenBSD 3.8 – Hackers of the Lost RAID

Evolução do OpenBSD

- OpenBSD 3.8 – Hackers of the Lost RAID
  - bioctl – Gerência de RAID
  - ipsecctl – Gerência IPSEC
  - ifstated – Monitorar Interface
  - sasyncd – Sincronismo IPSec
  - Novos hardwares
  - Mais de 3200 pacote pré compilados

Características do OpenBSD

- Focado na portabilidade, padronização (POSIX, ANSI, X/Open, etc), exatidão, segurança proativa e criptografia integrada

- Sua inspiração é ser o número um da indústria de segurança

- Desenvolvido por voluntários e os fundos são provenientes de CD's, camisetas e doações

Características do OpenBSD

- Até Junho de 2002, o site do OpenBSD mantinha o slogan "No remote hole in the default install, in nearly 6 years." Depois que uma falha foi descoberta no OpenSSH, foi alterado para "Only one remote hole in the default install, in more than 8 years".

Características do OpenBSD

- Impossibilidade de Buffer Overflow devido a novas tecnologias
  - strlcpy() e strlcat()
  - Memory protection purify (W^X, .rodata segment, Guard pages, Randomized malloc(), Randomized mmap(), atexit() and stdio protection)
  - Separação de Privilégio
  - Revogação de Privilégio
  - Cadeia Chroot
  - Novos uids
  - ProPolice

Características do OpenBSD

- Alertas de segurança OpenBSD 3.7
  - 07/06/05: Fix a buffer overflow, memory leaks, and NULL pointer dereference in cvs
  - 20/06/05: Fix a race condition in sudo
  - 06/07/05: Fix a buffer overflow in the zlib library
  - 21/07/05: Fix another buffer overflow in the zlib library

Características do OpenBSD

- Todos os sistemas operacionais modernos utilizam o código BSD em alguma parte de seu desenvolvimento

- A Internet esta apoiada na pilha TCP/IP que o time BSD ajudou a desenvolver sendo um advento revolucionário por ter incorporado a conexão diretamente no kernel possibilitando um aumento da performance e que os sistemas operacionais entrassem na era do “networking”

Características do OpenBSD

- O código está apoiado na licença BSD, realmente livre, que permite qualquer um fazer o que bem entender com o código, incluindo ganhar dinheiro licenciando o código e usá-lo em outro trabalho

Características do OpenBSD

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution

Características do OpenBSD

3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

> This product includes software developed by the University of California, Berkeley and its contributors

4. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

Filtro de Pacotes - PF

- Lançado como parte integrante do OpenBSD 3.0 em dezembro de 2001

- Necessário após mudança de licença do projeto IPFilter, que acompanhava a instalação base do OpenBSD

- Durante algumas semanas o OpenBSD ficou sem firewall na instalação padrão

Filtro de Pacotes - PF

- Normalização de Pacotes

Ajuda a evitar este tipo de ataque normalizando o tráfego passado em ambas as direções. A identificação de sistemas operacionais pode ser frustrada utilizando esta técnica provendo uma maior segurança em uma rede. Este tipo de característica ajuda a fazer o pf único, no meio de dispositivos de filtragem comerciais

Filtro de Pacotes - PF

- Limite de tráfego

  Filas Baseadas na Classe (CBQ)

  Filas de Prioridade (PRIO)

  Random Early Detection (RED)
    Derruba pacotes aleatórios

  Explicit Congestion Notification (ECN)
    Derruba pacote e avisa (cliente e servidor) que o pacote foi derrubado por congestionamento

Filtro de Pacotes - PF

- Balanço de Carga

Possibilidade de que links possam ser balanceados e roteando conexões entre eles, podendo haver um aumento de performance.

Filtro de Pacotes - PF

- Impressões digitais baseadas no SO

Baseado no p0f, reconhece assinaturas de sistemas operacionais. Útil para barrar acessos de worms, vírus e afins.

Filtro de Pacotes - PF

- KEEP STATE

Funciona com TCP, UDP e ICMP, mantendo a tabela de conexão e não processando as regras de um pacote caso faça parte da tabela de conexão

Filtro de Pacotes - PF

- MODULATE STATE

Mesmas características que o KEEP STATE, exceto que trabalha somente com TCP e o número de seqüência inicial é fortemente aleatório

Filtro de Pacotes - PF

- SYNPROXY STATE

Proxy de “HANDSHAKE” entre cliente e servidor destino. Elimina alguns tipos de ataque “TCP SYN flood”

Filtro de Pacotes - PF

- PFSYNC

Interface para sincronismo das tabelas de estado do PF

Common Address Redundancy Protocol(CARP)

- Protocolo que permite múltiplos hosts na mesma rede local compartilharem um mesmo endereço IP

- Funcionalidade existente no VRRP, porém o CARP difere em alguns aspectos

- É feito para prover grande segurança e é um protocolo independente (pode suportar tanto IPV4 como IPV6)

- Permite load balance em adicional a alta disponibilidade

Common Address Redundancy Protocol(CARP)

- Introduzido no OpenBSD 3.5 em Outubro de 2003
- Opção free para substituir o VRRP
- IPV4 e IPV6
- Proteção via SHA1
- Alta Disponiblidade

Common Address Redundancy Protocol(CARP)

- Layer 2 Load Balance (ARP)
- Intervalos de advertisement configuráveis (quem mais adverte vira master)
- Usado em conjunto com o PF e PFSYNC provê alta disponibilidade
- Portado para o FreeBSD em Fevereiro de 2005

Common Address Redundancy Protocol(CARP)

- Exemplos:

➔ /etc/hostname.carp0:
inet 10.0.0.1 255.255.255.0 10.0.0.255 vhid 1 pass minhasenha

➔ /etc/hostname.carp1:
inet 192.168.0.1 255.255.255.0 192.168.0.255 vhid 2 pass outrasenha

➔ /etc/hostname.carp0:
inet 10.0.0.1 255.255.255.0 10.0.0.255 vhid 1 advskew 100 pass minhasenha

➔ /etc/hostname.carp1:
inet 192.168.0.1 255.255.255.0 192.168.0.255 vhid 2 advskew 100 pass outrasenha

Common Address Redundancy Protocol(CARP)

- Exemplos:

➔ /etc/sysctl.conf:

```
net.inet.carp.allow=1
net.inet.carp.preempt=1
net.inet.carp.log=0
net.inet.carp.arpbalance=0
```

➔ Tcpdump:

```
13:14:01.287589 CARPv2-advertise 36: vhid=1 advbase=1
advskew=0 (DF)
13:14:02.371921 CARPv2-advertise 36: vhid=1 advbase=1
advskew=0 (DF)
13:14:03.470989 CARPv2-advertise 36: vhid=1 advbase=1
advskew=0 (DF)
13:14:04.568946 CARPv2-advertise 36: vhid=1 advbase=1
advskew=0 (DF)
```

Common Address Redundancy Protocol(CARP)

Considerações:

- O Load Balance não é perfeito. O CARP utiliza um Hash com o IP de origem para determinar qual das máquinas irá responder pela conexão.

- Como é um load balance baseado em Layer 2, uma das máquinas pode receber mais carga que a outra.

Quality of Services (QoS)

- ALTQ: Alternate Queueing for BSD UNIX
  - System FrameWork
  - Componentes de QoS
  - Ferramentas de Gerência

Quality of Services (QoS)

- O ALTQ funciona de três formas, que são divididas em:
  - FIFO
  - Filas Baseadas em Classe (CBQ)
  - Fila de Prioridade(PRIQ)

Quality of Services (QoS)

- CBQ (Filas Baseadas em Classe ou Class Based Queuing)
  - Divide a largura de banda em diversas filas ou classes
  - Endereço de origem ou destino, portas, protocolos, etc
  - Vários níveis de prioridade

Quality of Services (QoS)

- CBQ (Filas Baseadas em Classe ou Class Based Queuing)
  - Forma hierárquica de organização

Fila Raiz (10Mbps)
- Fila ssh (2Mbps, priority 1)
- Fila http (5Mbps, priority 4)
- Fila mail (2Mbps, priority 5)
- Fila ftp (1Mbps, priority 7)

Quality of Services (QoS)

- PRIQ (Fila de Prioridade ou Priority Queuing)
  - Uma fila com alta prioridade é "*sempre*" processada na frente de uma fila com prioridade menor
  - A fila raiz é definida onde é configurado o total de banda disponível e, então, subfilas são definidas sob a raiz

Quality of Services (QoS)

- PRIQ (Fila de Prioridade ou Priority Queuing)

  Fila Raiz (10Mbps)
  - Fila ssh (priority 1)
  - Fila http (priority 4)
  - Fila mail (priority 5)
  - Fila ftp (priority 7)

Quality of Services (QoS)

- RED (Random Early Detection)
  - Seu trabalho é evitar congestionamento na rede certificando-se de que a fila nunca fique cheia
  - Calcula o tamanho médio da fila
  - Valor mínimo e um valor máximo

Quality of Services (QoS)

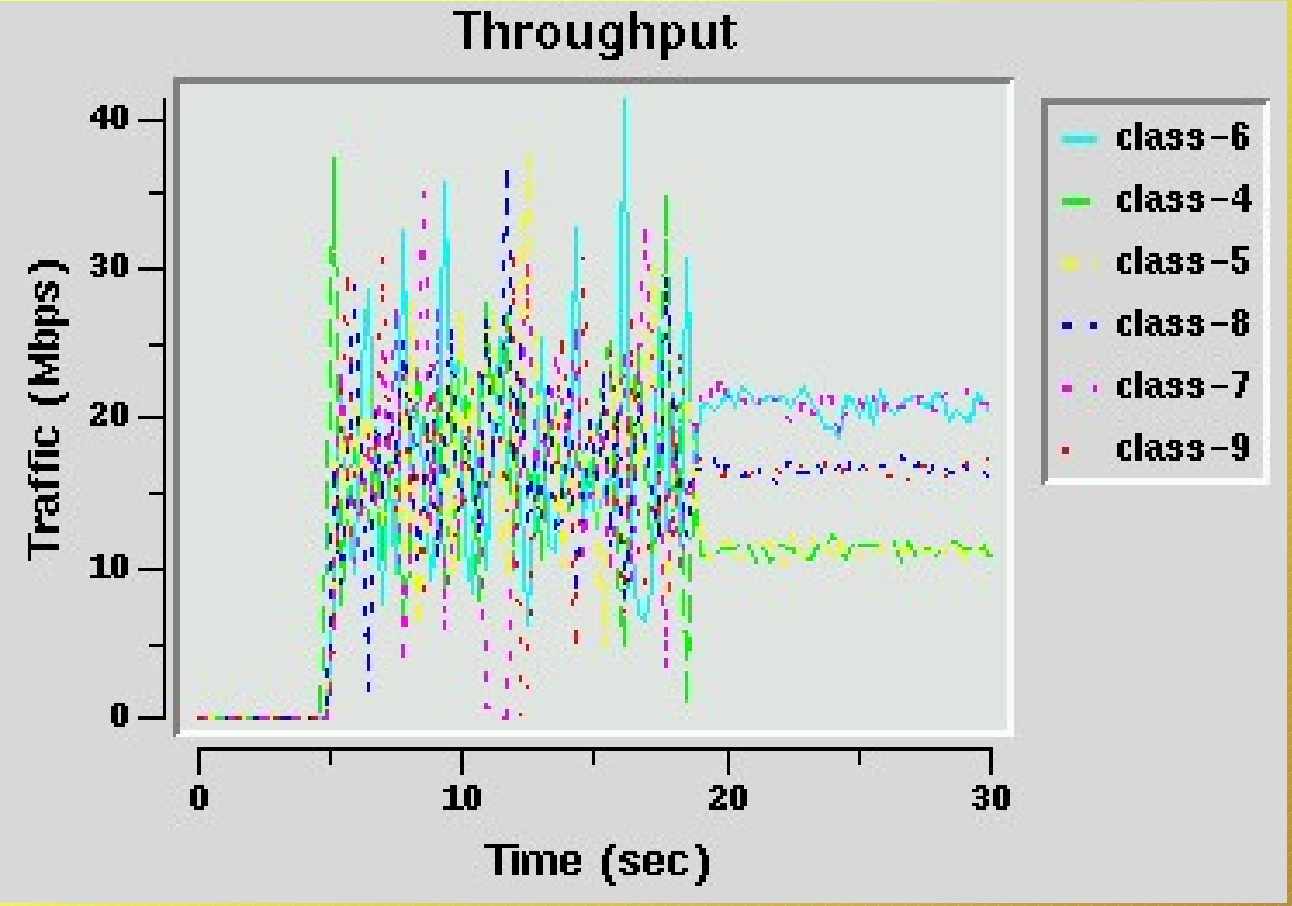
- RED (Random Early Detection)
  - Quando o tamanho médio da fila se aproxima do valor máximo, mais pacotes são descartados
  - Escolhe aleatoriamente de quais conexões ele irá descartar os pacotes
  - Somente TCP e nunca tráfego UDP ou ICMP

Quality of Services (QoS)

- ECN (Explicit Congestion Notification)
  - Trabalha em conjunto com RED
  - Insere uma flag no cabeçalho do pacote ao invés de descartá-lo

Quality of Services (QoS)

- Exemplo de Tráfego



Quality of Services (QoS)

- Exemplo de Regras

```
altq on dc0 scheduler cbq bandwidth 10Mb queue { std,
http, mail, ssh }
    queue   std bandwidth 10% cbq(default)
    queue   http bandwidth 60% priority 2 cbq(borrow red) {
employees, developers }
    queue    developers bandwidth 75% cbq(borrow)
    queue    employees bandwidth 15%
    queue    mail bandwidth 10% priority 0 cbq(borrow ecn)
    queue    ssh bandwidth 20% cbq(borrow)
{ ssh_interactive, ssh_bulk }
    queue    ssh_interactive bandwidth 100% priority 7
    queue    ssh_bulk bandwidth 100% priority 0
```

Virtual Private Network (VPN)

- L2TP (Layer 2 Tunneling Protocol)
- PPTP (Point-to-Point Tunneling Protocol)
- MPLS
- SSH
- IPSEC

Virtual Private Network (VPN)

- IPSEC
  - É uma extensão do protocolo IP
  - Serviços básicos
    - Autenticação: garantir que os dados são quem eles realmente dizem ser
    - Verificação: garantir que os dados não foram alterados
    - Confidencialidade: garantir que os dados não podem ser entendidos, mesmo se vistos

Virtual Private Network (VPN)

- IPSEC
  - Autentication Header (AH) – Protocolo 51
    - Garantir a autenticidade do pacote
    - HMAC, MD5, SHA1, SHA2, etc
  - Encapsulation Security Payload (ESP) – Protocolo 50
    - Garantir a confidencialidade dos dados
    - DES, Blowfish, 3-DES, CAST, AES, etc

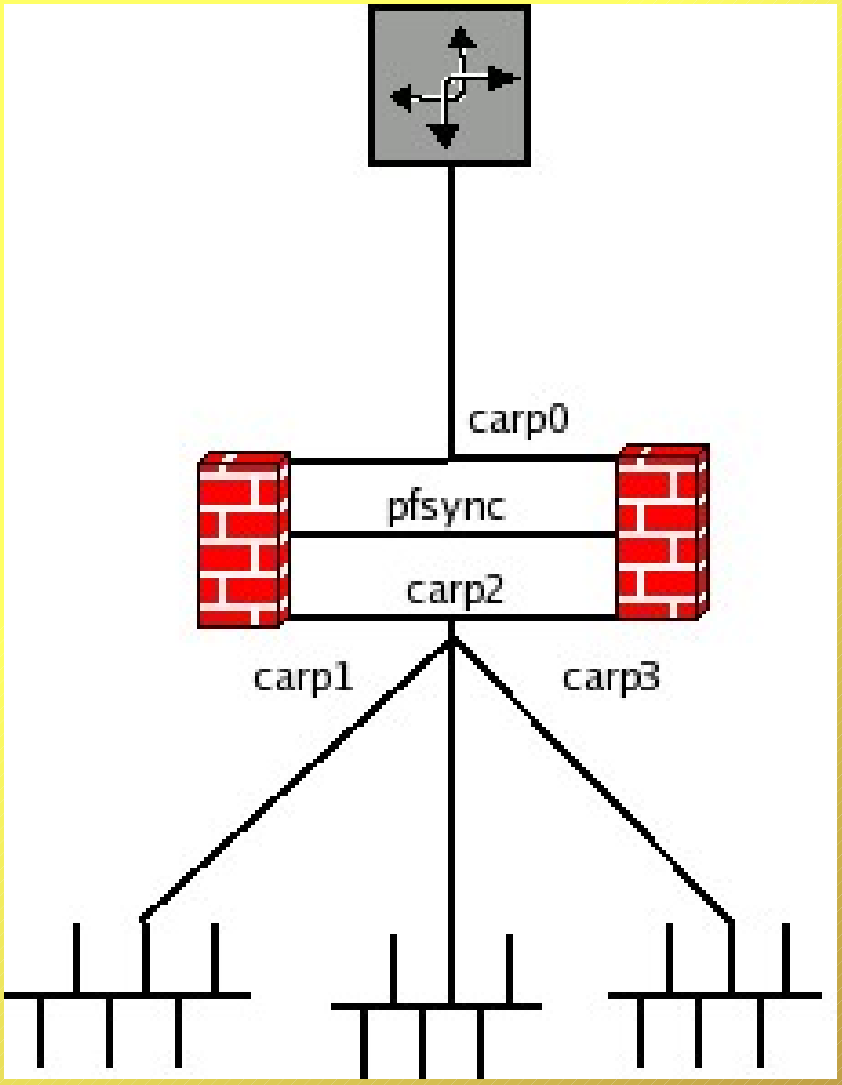
Virtual Private Network (VPN)

- IPSEC
  - SA (Security Association)
    - Definir métodos de segurança que serão aplicados aos pacotes
  - ISAKMP
    - Uma SA pode ser configurada dinamicamente ou manualmente
    - ISAKMP define como as entidades instituirão um canal de comunicação seguro

Estudo de Caso

- Substituição de Firewall comercial
- Necessidade de migração de hardware
- Não teria Alta disponibilidade
- Software Proprietário
- Custo elevado !!

Estudo de Caso



Estudo de Caso

- Regras QoS:

```
altq on { em0 } cbq bandwidth 100Mb queue { deflt,
              http, ssh, mail, radius, bancos, dns, rsets }
queue   deflt bandwidth 22% priority 7  cbq(default)
queue   http bandwidth 30% priority 4
queue   mail bandwidth 30% priority 4
queue   dns bandwidth 4% priority 2
queue   ssh bandwidth 5% priority 1
queue   radius bandwidth 3% priority 1
queue   bancos bandwidth 4% priority 3
queue   rsets bandwidth 2% priority 7
```

Estudo de Caso

- Regras Firewall:

```
#Protecoes anti spoof
block return-icmp in log  quick inet  from
<invalidos>  to any  label "REJECT "
block return-icmp   out  log  quick inet  from
<invalidos>  to any  label "REJECT "

#Bloqueio de spam
block return-icmp in log quick inet from <spam>
to any label "SPAM "
```

Estudo de Caso

- Regras Firewall:

```
#OSPF
pass in proto 89 from <ospfnet> keep state
pass out proto 89 from <ospfnet> keep state
pass in from 224.0.0.0/4 keep state
pass out from 224.0.0.0/4 keep state

#SSH para o FW e zebra/ospfd para o carp
pass in quick log on em0 proto tcp from <admin> to
<carp0>port 2600:26006 keep state  label  "Zebra
REMOTO" queue ssh
pass in quick log on em0 proto tcp from <admin> to
<fw_ip> port ssh keep state  label  "SSH REMOTO"
queue ssh
```

Estudo de Caso

- Status das Conexões (pfctl -ss -v):

self tcp 200.195.198.7:80 <- 201.35.2.1:50547
TIME_WAIT:TIME_WAIT
   [3086304581 + 7340] [14890543 + 7777]
   age 00:01:18, expires in 00:00:30, 16:23 pkts, 1759:27752
bytes, rule 71

self tcp 200.195.192.45:80 <- 198.81.8.1:60868
FIN_WAIT_2:FIN_WAIT_2
   [608169301 + 14600] wscale 0 [3686070616 + 16332]
wscale 0
   age 00:01:04, expires in 00:00:29, 9:8 pkts, 1560:3673
bytes, rule 222
self tcp 200.195.192.45:80 <- 198.81.9.1:55170
FIN_WAIT_2:FIN_WAIT_2

Estudo de Caso

- Status das Conexões (pfctl -ss -v):

self tcp 200.195.198.2:80 <- 200.138.65.1:9699
FIN_WAIT_2:ESTABLISHED
   [938652815 + 17520] [3507662902 + 17520]
   age 00:00:18, expires in 00:00:43, 20:24 pkts, 4635:24902
bytes, rule 64

self tcp 200.195.198.3:80 <- 200.138.65.1:9702
ESTABLISHED:ESTABLISHED
   [500506525 + 17107] [3508539066 + 6432]
   age 00:00:15, expires in 00:59:46, 4:3 pkts, 491:541 bytes,
rule 69

Estudo de Caso

- Status das Conexões (pfctl -ss -v):

self tcp 200.101.160.44:1923 -> 200.195.198.2:80
ESTABLISHED:SYN_SENT
   [383883491 + 5840] [1143059736 + 16384]
   age 00:00:05, expires in 00:00:29, 2:3 pkts, 96:132
bytes, rule 2

self tcp 200.103.255.44:3275 -> 200.195.198.2:80
ESTABLISHED:CLOSING
   [1823968355 + 6700] [780476401 + 5206]

Estudo de Caso

- Status CBQ

queue deflt bandwidth 18Mb priority 4 cbq(default)
[pkts: 9918766 bytes: 1492078565 dropped pkts: 243 bytes: 351362]
[qlength: 0/ 50 borrows: 0 suspends: 30061]
[measured: 302.2 packets/s, 173.72Kb/s]

queue http bandwidth 31Mb priority 4 cbq(red borrow)
[pkts: 79334795 bytes: 75182837191 dropped pkts: 548 bytes: 506760]
[qlength: 0/ 50 borrows: 1108447 suspends: 2116]
[measured: 1364.5 packets/s, 9.60Mb/s]

queue mail bandwidth 25Mb priority 4 cbq(red borrow)
[pkts: 38467710 bytes: 25814183190 dropped pkts: 301 bytes: 222802]
[qlength: 0/ 50 borrows: 103335 suspends: 364]
[measured: 610.4 packets/s, 2.96Mb/s]

Estudo de Caso

- Status das Regras

pass in quick on em0 inet proto tcp from any to 200.200.200.200 port = www flags S/SA keep state queue http
  [Evaluations: 4288260   Packets: 19199408 Bytes: 10401168886 States: 937   ]

pass in quick on em0 inet proto tcp from any to 200.200.200.200 port = https flags S/SA keep state queue http
  [Evaluations: 26526   Packets: 3350   Bytes: 1060230 States: 0   ]

pass in quick on em0 inet proto tcp from any to 200.200.200.200 port = ftp-data flags S/FSRPAU modulate state queue http
  [Evaluations: 26365   Packets: 0   Bytes: 0   States: 0
   ]

Contato

Através do site ou e-mail

http://web.onda.com.br/humberto
http://web.onda.com.br/nadal

humberto@onda.com.br
nadal@onda.com.br

Créditos

Sites consultados:

- OpenBSD
http://www.openbsd.org

- Site Pessoal João Henrique F. Freitas
http://paginas.terra.com.br/informatica/joaohf/openbsd/openbsd.html

Créditos

- Figura Slide 1:
http://www.openbsd.org/art/ramblo.jpg

- Figura Slide 12:
http://www.openbsd.org/images/Rock.jpg

- Figura Slide 14:
http://www.openbsd.org/images/Systemagic.jpg

- Figura Slide 16:
http://www.openbsd.org/images/MrPond.gif

- Figura Slide 18:
http://www.openbsd.org/images/Barbarian.gif

- Figura Slide 20:
http://www.openbsd.org/images/Hood.gif

- Figura Slide 22:
http://www.openbsd.org/images/Carp.gif

Créditos

- Figura Slide 24:
http://www.openbsd.org/images/Ponderosa.jpg

- Figura Slide 26:
http://www.openbsd.org/images/Wizard.jpg

- Figura Slide 28:
http://www.openbsd.org/images/Jones.jpg

- Figura Slide 62:
http://www.csl.sony.co.jp/person/kjc/software/cbq.gif

- Figura Slide 69:
  Arquivo Pessoal